


# Austin Animal Center Storyboard/Outline

Project by: Project-A-Team





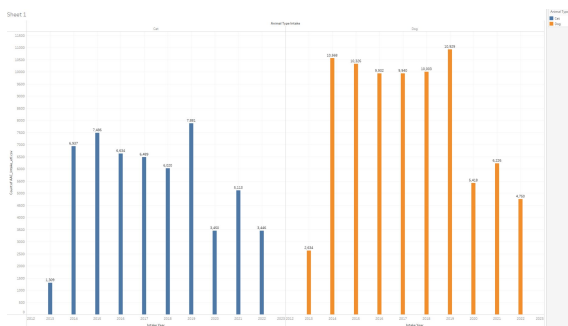
Objective of Analysis: How to improve adaptability of animals in the shelter.



Adopt



- Current issue: Overcrowding at Austin Animal Center.
  - Currently restricting animal intakes starting September 13th.
  - AAC Has over 700 animals.
  - <https://www.kvue.com/article/life/animals/austin-animal-center-temporarily-restricts-animal-intakes/269-9fbf7f68-8682-4099-ad9c-7f84bc48c8c6?fbclid=IwAR284aVvc8VDkzbKxpnA6VFfa6OXDIg2gRj82cYLNG2IEQm-nq1fyq3i9Y>
- Graphic for yearly intakes of Cats vs Dogs throughout the years.





Overview of Analysis: What we want to solve with data storytelling.





Questions to be answered (picking our top 5 soon, waiting for our data to tell the story):

- 1). What month has most intakes/outcomes
- 2). Distribution of dog and cats.
- 3). Where in Austin are most intakes coming from?
- 4.) Average dog and cat intake in a week?
- 5). Average time in the shelter by age ?



# Blueprint of DashBoard



- 1). What area of Austin is most likely to have strays? We will use a layered map in our dashboard to easily show common areas for strays.
- 2). What month has most intakes? Using bar graph to show month to month basis of intakes.
- 3). What day of the week is most likely to have adoptions? Using bar graph for days of the week to story tell what days are best for adoptions.
- 4). Where in Austin are most intakes coming from? Using a map to determine where most intakes are coming from.
- 5). Average time in shelter by age? Bar graph to show relationship between age and time.



Description of tools that will be used to  
create final dashboard

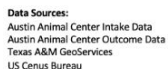




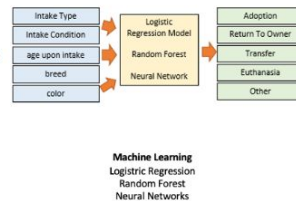
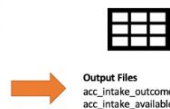
- We will be using tableau to create our final dashboard and will be creating our interactive elements through tableau.
- Interactive elements:
  - Filters for different years, breeds, age, maps with layers for how location is a factor, and among other interactive elements being discussed in our group.



# Project Workflow - Database & Machine Learning



PostgreSQL13  
Austin Animal Center Intake Data  
Austin Animal Center Outcome Data  
Texas A&M GeoServices Lat and Long  
US Census Bureau





# Data Analysis



The data analysis process began with the consideration of two datasets from Austin Animal Center.


- *Austin\_Animal\_Center\_Intakes.csv*

Intakes represent the status of animals as they arrive at the Animal Center. All animals receive a unique Animal ID during intake. Annually over 90% of animals entering the center, are adopted, transferred to rescue or returned to their owners.

- *Austin\_Animal\_Center\_Outcomes.csv*

The Outcomes data set reflects that Austin, TX. is the largest "No Kill" city in the country. All animals received with unique Animal IDs are safely adopted or transferred and this data represents that.

- Intake data and Outcome data contains 12 columns with 144k rows.



To analyze the Successful/Unsuccessful outcome of Adoption or relocation of pets based on data

✓ Description of preliminary data preprocessing

For preprocessing, we needed to clean up entries from each dataset and to get rid of bad or erroneous data. The columns that were entered as Strings were converted to numeric/date fields wherever possible so that the dataset is helpful and contains numeric data which will be useful for Machine Learning models. Next in the database we merged the intake and outcome datasets based on unique identifier (as described in the Database section) for further analysis.

✓ Description of preliminary feature engineering and preliminary feature selection, including their decision-making process

1. AAC\_Intake\_etl\_step1.ipynb

1. The "Age upon Intake" column was a string containing (days, weeks, years etc.), the column was split and then "Age Upon Intake(days)" & "Age Upon Intake(years)" was calculated.
2. Using DateTime Series Intake Month, Intake year and Intake Weekday & Intake Hour are calculated.
3. Dropping the name column.
4. Created a new column - Intake Frequency as in how many times a same animal with unique Animal ID is brought to AAC.
5. Cumulative frequency is #4 is calculated for each row based on the Intake Time and date and time in Ascending order.
6. New transformed intake\_df with desired columns is put in a csv file and using postgres sql connection is loaded in the database.

## 2. AAC\_Outcome\_etl\_step1.ipynb

All the above steps are repeated for the outcome data (with outgoing animal data)

## 3. Database Integration Description

1. Postgres SQL database was used for storing and structuring the data.
2. The files produced in the above steps were extracted using sqlalchemy to export the data into an AAC database into tables intake\_df, outcome\_df and zipcodes\_df.


## Create a connection to Postgres using sqlalchemy

```
#read csv file
outcome_df = pd.read_csv("../Resources/AAC_Outcome_etl.csv")

#Create a connection string for PostgreSQL
#"postgresql://[user]:[password]@[location]:[port]/[database]"
db_string = f"postgresql://postgres:{db_password}@127.0.0.1:5432/AAC"

#create a database engine
engine = create_engine(db_string)

outcome_df.to_sql(name='outcome_df', con=engine, if_exists='replace')
```

- 
- Once loaded into Postgres SQL tables, the files were altered to create primary key off the ("animal\_id\_intake", order\_of\_intake") and ("animal\_id\_outcome", order\_of\_outcome") as well as altering the date fields. The zipcode table was altered to create a primary key of "index\_id" that joins to the index\_id\_intake.
  - Based on the new primary keys made up of a combination of Compound key from intake\_df (animal\_id\_intake & order\_of\_intake) and outcome\_df (animal\_id\_outcome & order\_of\_outcome), the tables are joined along with zipcode to get a combined dataset containing both data together.
  - Using case statements columns were split up and restructured for analysis such as subtypes for breeds based on the predominate identified breed, if it contained Pit Bull and date calculations.
  - The acc\_intake\_outcome and acc\_intake\_available were then exported to "acc\_intake\_outcome.csv" & "acc\_intake\_available.csv" and also connected to the machine learning script using sqlalchemy.

```
M from sqlalchemy import create_engine
from config import db_password
```

```
M # Create a connection with the database in postgres
db_string = f"postgresql://postgres:{db_password}@127.0.0.1:5432/AAC"
```

```
M engine = create_engine(db_string)
```

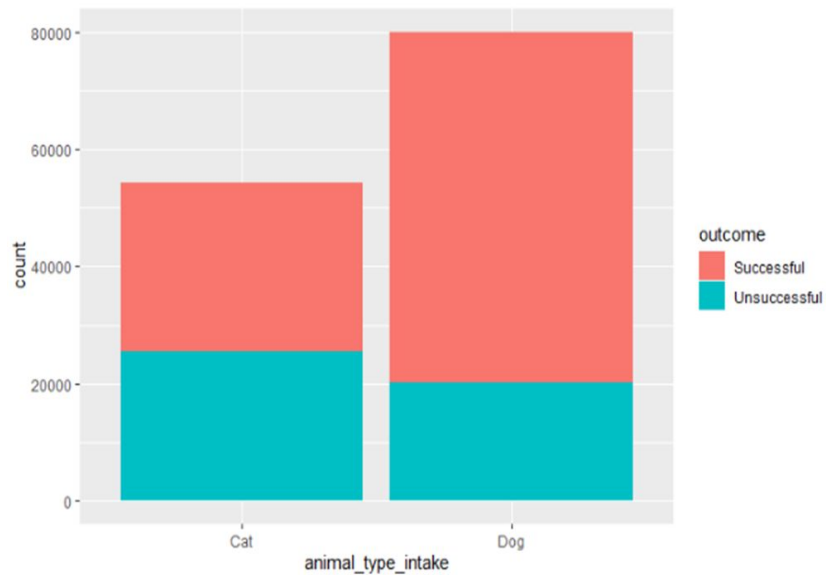
```
M # read the table from the database
df = pd.read_sql_table("acc_intake_outcome", engine)
df.head()
```

```
1:  index_id_intake  animal_id_intake  datetime_intake  monthyear_intake  found_location  intake_type  intake_condition  animal_type_intake  sex_upon_intake
0         70641         A178569      2014-03-17
   09:45:00      March 2014      Austin (TX)      Public
   Assist      Normal      Dog      Neutered Male
```

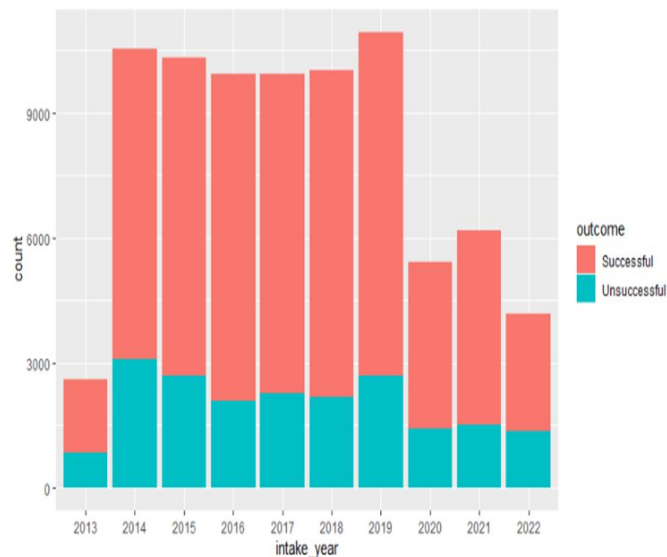
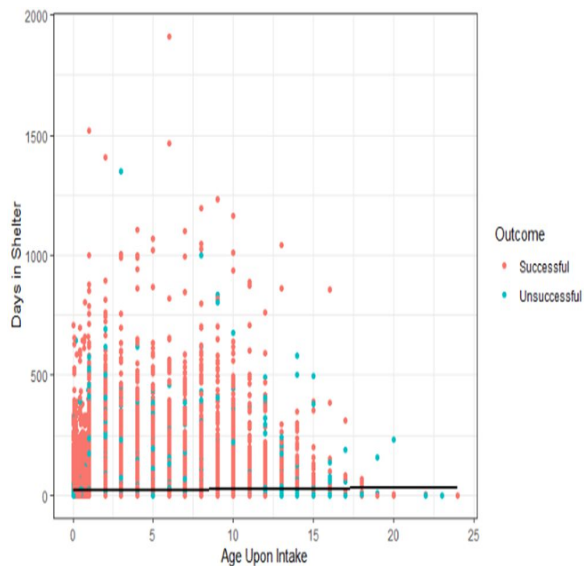




## Analyzing Combined Dataset - Filtering for dogs and cats only

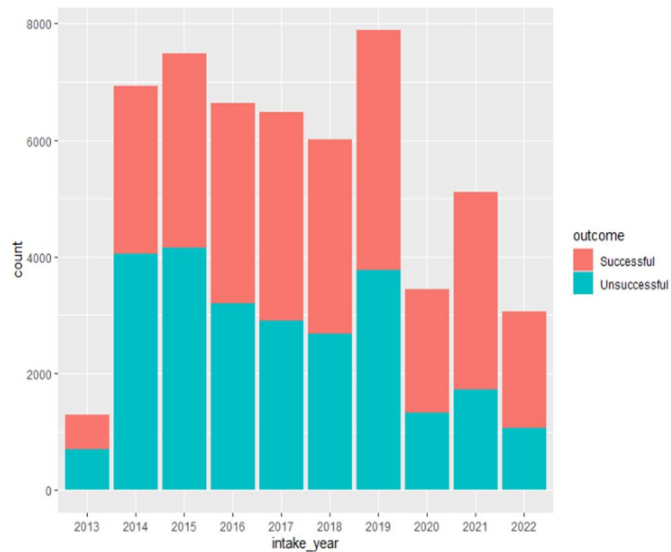
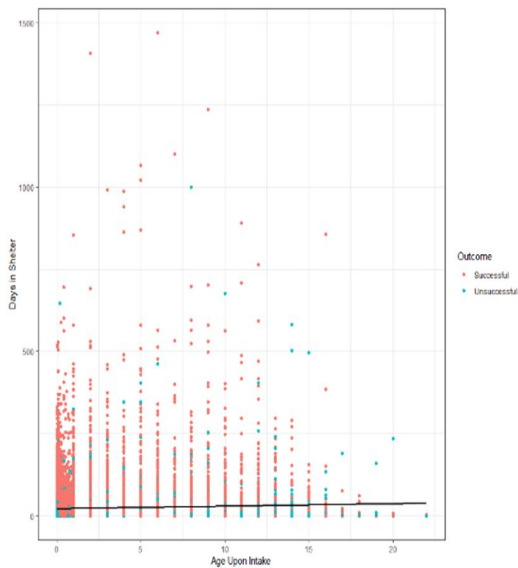


## Dogs Categorized As Successful/Unsuccessful Outcome






## Cats Categorized As Successful/Unsuccessful Outcome





# Machine Learning Model



Code -

[github](#) - AAC\_ML\_model\_Adopted(Success\_Failure)-Dog.ipynb


[github](#) - AAC\_ML\_model\_Adopted(Success\_Failure)-Cat.ipynb &

To analyze the Successful outcome of Adoption or relocation of an Animal , we separated the datasets into 2 main category of Animals - Dogs & Cats.

To run ML model we took consideration of following categories-

1. Age
2. Breed
3. Color
4. Intake type
5. Intake condition
6. Outcome type

For Dog -

- 
1. df\_dog contains the entire data from “acc\_intake\_outcome.csv” for “Dog”
  2. `df_dog = df[df['animal_type_intake']=='Dog']`
  3. We checked for Unique values of all categories  
We checked for Unique values of all categories

For Cat -

1. df\_cat contains the entire data from “acc\_intake\_outcome.csv” for “Cat”
2. `df_cat = df[df['animal_type_intake']=='Cat']`
3. We checked for Unique values of all categories  
We checked for Unique values of all categories

1. Hot encoding color\_intake , breed\_intake, intake\_type by getting the top 10 frequently occurring kinds for each of them and then merged into the main dataframe df\_dog\_ML. Like

```
color_intake_counts.head(10)
```

Black/White	9464
Brown/White	4577
White	4261
Tan/White	4204
Black	4195
Tan	3526
Brown	3208
Black/Tan	3070
Tricolor	3068
Black/Brown	2895

Name: color\_intake, dtype: int64

```
breed_intake_counts.head(10)
```

Pit Bull Mix	5332
Labrador Retriever Mix	4414
Chihuahua Shorthair Mix	3859
German Shepherd Mix	2007
Pit Bull	1212
Australian Cattle Dog Mix	1043
Labrador Retriever	830
Chihuahua Shorthair	821
German Shepherd	696
Dachshund Mix	687

Name: breed\_intake, dtype: int64

```
intake_type_count
```

Stray	10984
Owner Surrender	3599
Public Assist	1386
Abandoned	85
Euthanasia Request	33

Name: intake\_type, dtype: int64

2. intake\_condition is categorized into three main categories - Normal , Aged & Other - and then they are hot encoded and merged into the main dataframe df\_dog\_ML

```
intake_condition_normal = ['Normal', 'Behavior']|
intake_condition_aged = ['Aged']
intake_condtion_other = ['Injured', 'Sick', 'Nursing', 'Neonatal', 'Other', 'Medical', 'Feral', 'Pregnant', 'Med Urgent']
```

3. outcome\_type is categorized into two main categories - Success & Failure - and then they are hot encoded and merged into the main dataframe df\_dog\_ML.

```
intake_condition_normal = ['Normal', 'Behavior']|
intake_condition_aged = ['Aged']
intake_condtion_other = ['Injured', 'Sick', 'Nursing', 'Neonatal', 'Other', 'Medical', 'Feral', 'Pregnant', 'Med Urgent']
```



3. outcome\_type is categorized into two main categories - Success & Failure - and then they are hot encoded and merged into the main dataframe df\_dog\_ML.

```
: ## Determine which values to replace
# replace_intake_condition = list(intake_type[intake_type < 465].index)

other_outcome_type_list = ['Transfer', 'Euthanasia', 'Died', 'Disposal', 'Missing']
success_outcome_list = ['Adoption', 'Return to Owner', 'Rto-Adopt']

## Replace in DataFrame
for outcome in other_outcome_type_list:
    df_dog_ML.outcome_type = df_dog_ML.outcome_type.replace(outcome, "Failure")

for outcome in success_outcome_list:
    df_dog_ML.outcome_type = df_dog_ML.outcome_type.replace(outcome, "Success")

## Check to make sure binning was successful
df_dog_ML.outcome_type.value_counts()
```

Then we used the function get\_dummies on df\_dog\_ML["outcome\_type"] and then merging into the dataframe df\_dog\_ML.

✓ Description of how data was split into training and testing sets

The data is split into X and y. Where below are the features -

X = The hot encoded values of the following features

'age\_upon\_intake(days)'

'age\_upon\_outcome(days)'

'days\_in\_shelter'

'intake\_condition'

'color\_intake'

'breed\_intake'

'intake\_type'

Y = 'outcome\_type' encoded for Success and Failure.

```
# Seperate the features X from the target Y
```

```
y = df_dog_ML.Success
```

```
columns=["Success", "Failure"]
```

```
X = df_dog_ML.drop(columns=columns)
```

```
# Split training/test datasets
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=42, stratify=y)
```

- ✓ Explanation of model choice, including limitations and benefits.

We used Logistic Regression and Random Forest Classification models to analyze the data.

Logistic Regression - is performed when we are expecting a Binary Outcome - Here we are running the ML model to determine if Dog / Cat will have success or Failure as outcome for given categories or features in consideration.

Random Forest Classification - This model produces good predictions, and is capable to handle large datasets efficiently. This model helps in producing higher level of accuracy. Below is the Confusion Matrix for Dogs and Cats.

```
# Displaying results
print("Confusion Matrix for Dogs")
display(cm_df)
print(f"Accuracy Score : {acc_score}")
print("Classification Report")
print(classification_report(y_test, predictions))
```

Confusion Matrix for Dogs

	Failure	Success
Failure	27	565
Success	62	1732

Accuracy Score : 0.7372170997485331

Classification Report

	precision	recall	f1-score	support
0	0.30	0.05	0.08	592
1	0.75	0.97	0.85	1794
accuracy			0.74	2386
macro avg	0.53	0.51	0.46	2386
weighted avg	0.64	0.74	0.66	2386

```
# Displaying results
print("Confusion Matrix for Cats.")
display(cm_df)
print(f"Accuracy Score : {acc_score}")
print("Classification Report")
print(classification_report(y_test, predictions))
```

Confusion Matrix for Cats.

	Failure	Success
Failure	79	67
Success	82	76

Accuracy Score : 0.5098684210526315

Classification Report

	precision	recall	f1-score	support
0	0.49	0.54	0.51	146
1	0.53	0.48	0.50	158
accuracy			0.51	304
macro avg	0.51	0.51	0.51	304
weighted avg	0.51	0.51	0.51	304



# Database

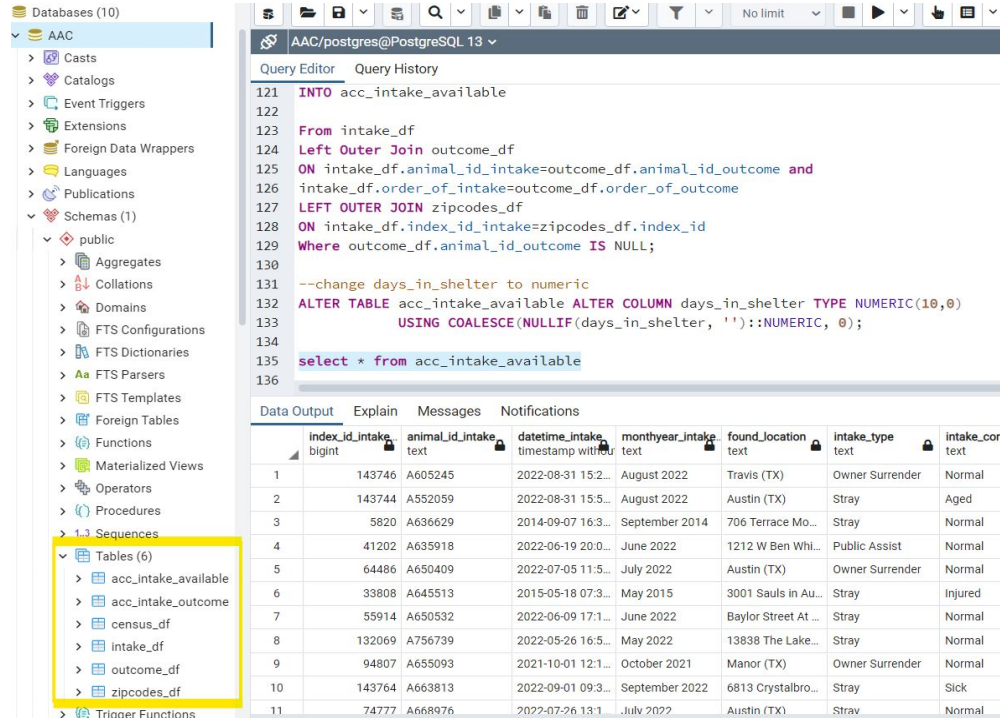


# Database Integration

*For this project, we utilized PostgresSQL and fully integrated the database into our project.*

- Database stores static data for use during the project
  - *Database stores multiple data tables used for compiling the final dataset.*
- Database interfaces with the project in some format (e.g., scraping updates the database)
  - *Database interfaces with the project using Jupyter Notebook Pandas and sqlalchemy to export the dfs into the Postgres AAC database and create the tables and to import the final sql table back to Jupyter Notebooks to use for the machine learning.*
- Includes at least two tables (or collections, if using MongoDB)
  - *The AAC database includes three tables used for the final dataset.*
- Includes at least one join using the database language (not including any joins in Pandas)
  - *SQL is used to join the three tables together and perform data manipulation on number, character and date columns.*
- Includes at least one connection string (using SQLAlchemy or PyMongo)
  - *Three connection strings using SQLAlchemy export the data into the Postgres database and two connection strings import the final dataset into the Jupyter Notebook machine learning scripts.*

# Postgres Database Stores Data Tables



The screenshot displays a PostgreSQL database management interface. On the left, a sidebar shows the database structure, including a schema named 'public' containing several tables. The 'Tables (6)' section is highlighted, showing tables like 'acc\_intake\_available', 'acc\_intake\_outcome', 'census\_df', 'intake\_df', 'outcome\_df', and 'zipcodes\_df'. The main area shows a SQL query editor with a query that updates the 'acc\_intake\_available' table. Below the query editor, a 'Data Output' tab displays a table with 11 rows of data.

**Query Editor:**

```
121 INTO acc_intake_available
122
123 From intake_df
124 Left Outer Join outcome_df
125 ON intake_df.animal_id_intake=outcome_df.animal_id_outcome and
126 intake_df.order_of_intake=outcome_df.order_of_outcome
127 LEFT OUTER JOIN zipcodes_df
128 ON intake_df.index_id_intake=zipcodes_df.index_id
129 Where outcome_df.animal_id_outcome IS NULL;
130
131 --change days_in_shelter to numeric
132 ALTER TABLE acc_intake_available ALTER COLUMN days_in_shelter TYPE NUMERIC(10,0)
133 USING COALESCE(NULLIF(days_in_shelter, '')::NUMERIC, 0);
134
135 select * from acc_intake_available
136
```

**Data Output:**

	index_id_intake	animal_id_intake	datetime_intake	monthyear_intake	found_location	intake_type	intake_con
	bigint	text	timestamp without time zone	text	text	text	text
1	143746	A605245	2022-08-31 15:2...	August 2022	Travis (TX)	Owner Surrender	Normal
2	143744	A552059	2022-08-31 15:5...	August 2022	Austin (TX)	Stray	Aged
3	5820	A636629	2014-09-07 16:3...	September 2014	706 Terrace Mo...	Stray	Normal
4	41202	A635918	2022-06-19 20:0...	June 2022	1212 W Ben Whi...	Public Assist	Normal
5	64486	A650409	2022-07-05 11:5...	July 2022	Austin (TX)	Owner Surrender	Normal
6	33808	A645513	2015-05-18 07:3...	May 2015	3001 Sauls in Au...	Stray	Injured
7	55914	A650532	2022-06-09 17:1...	June 2022	Baylor Street At ...	Stray	Normal
8	132069	A756739	2022-05-26 16:5...	May 2022	13838 The Lake...	Stray	Normal
9	94807	A655093	2021-10-01 12:1...	October 2021	Manor (TX)	Owner Surrender	Normal
10	143764	A663813	2022-09-01 09:3...	September 2022	6813 Crystalbro...	Stray	Sick
11	74777	A668976	2022-07-26 13:1...	July 2022	Austin (TX)	Stray	Normal

## Database Interfaces With AAC\_ML\_model\_Adopted(Success\_Failure) - Dog/Cat

```
In [2]: import pandas as pd
import sklearn as skl
from sklearn.preprocessing import OneHotEncoder
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestClassifier
```

```
In [3]: from sqlalchemy import create_engine
from config import db_password
```

```
In [4]: # Create a connection with the database in postgres
db_string = f"postgresql://postgres:{db_password}@127.0.0.1:5432/AAC"
```

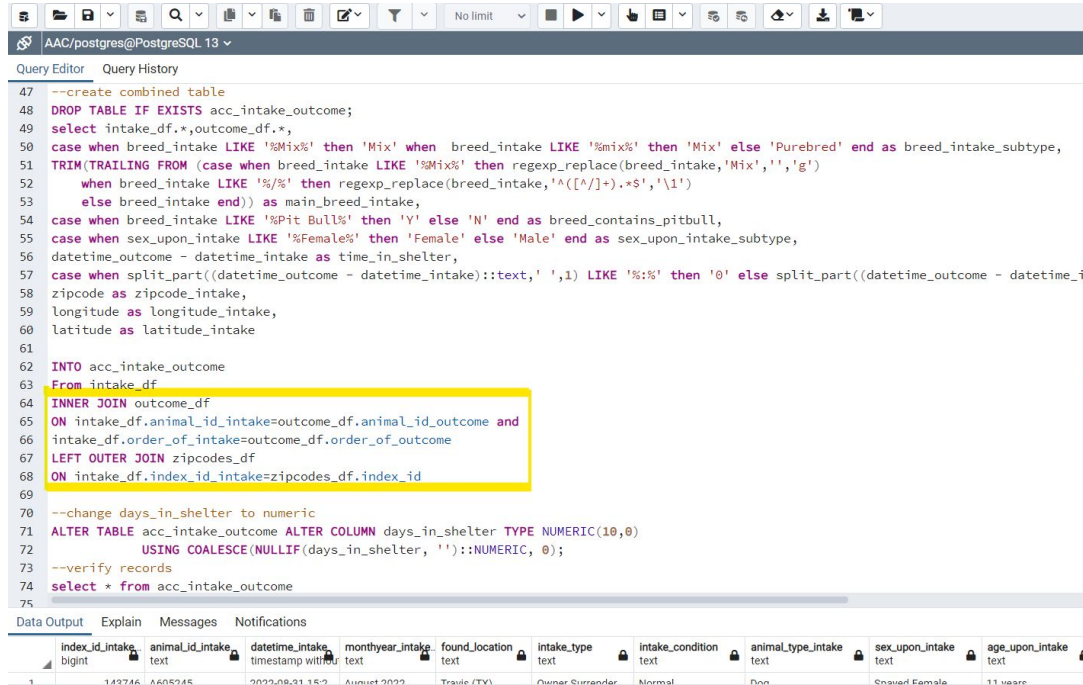
```
In [5]: engine = create_engine(db_string)
```

```
In [6]: # read the table from the database
df = pd.read_sql_table("acc_intake_outcome", engine)
df.head()
```

```
Out[6]:
```

	index_id_intake	animal_id_intake	datetime_intake	monthyear_intake	found_location	intake_type	intake_condition	animal_type_intake	sex_upon_intake
0	70641	A178569	2014-03-17 09:45:00	March 2014	Austin (TX)	Public Assist	Normal	Dog	Neutered Male
1	944	A287017	2015-08-16 12:19:00	August 2015	6620 Deatonhill Dr in Austin (TX)	Stray	Normal	Dog	Spayed Female
2	69127	A293383	2018-03-18 18:17:00	March 2018	6005 Walnut Hills in Austin (TX)	Stray	Sick	Cat	Neutered Male

# Database Includes Joins. Refer to full SQL @ /Database/aac.sql



```
47 --create combined table
48 DROP TABLE IF EXISTS acc_intake_outcome;
49 select intake_df.*,outcome_df.*,
50 case when breed_intake LIKE '%Mix%' then 'Mix' when breed_intake LIKE '%mix%' then 'Mix' else 'Purebred' end as breed_intake_subtype,
51 TRIM(TRAILING FROM (case when breed_intake LIKE '%Mix%' then regexp_replace(breed_intake, 'Mix', '', 'g')
52 when breed_intake LIKE '%%' then regexp_replace(breed_intake, '^([/]+).*$', '\1')
53 else breed_intake end)) as main_breed_intake,
54 case when breed_intake LIKE '%Pit Bull%' then 'Y' else 'N' end as breed_contains_pitbull,
55 case when sex_upon_intake LIKE '%Female%' then 'Female' else 'Male' end as sex_upon_intake_subtype,
56 datetime_outcome - datetime_intake as time_in_shelter,
57 case when split_part((datetime_outcome - datetime_intake)::text, ' ', 1) LIKE '%%' then '0' else split_part((datetime_outcome - datetime_i
58 zipcode as zipcode_intake,
59 longitude as longitude_intake,
60 latitude as latitude_intake
61
62 INTO acc_intake_outcome
63 From intake_df
64 INNER JOIN outcome_df
65 ON intake_df.animal_id_intake=outcome_df.animal_id_outcome and
66 intake_df.order_of_intake=outcome_df.order_of_outcome
67 LEFT OUTER JOIN zipcodes_df
68 ON intake_df.index_id_intake=zipcodes_df.index_id
69
70 --change days_in_shelter to numeric
71 ALTER TABLE acc_intake_outcome ALTER COLUMN days_in_shelter TYPE NUMERIC(10,0)
72 USING COALESCE(NULLIF(days_in_shelter, '')::NUMERIC, 0);
73 --verify records
74 select * from acc_intake_outcome
75
```

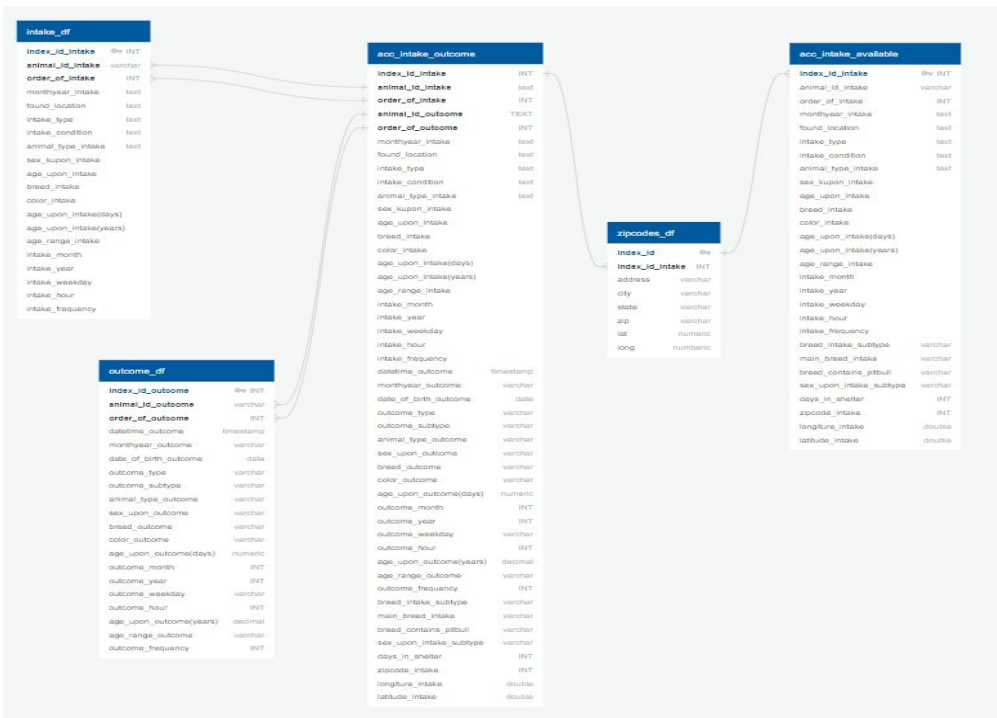
Data Output Explain Messages Notifications

	index_id_intake bigint	animal_id_intake text	datetime_intake timestamp without time zone	monthyear_intake text	found_location text	intake_type text	intake_condition text	animal_type_intake text	sex_upon_intake text	age_upon_intake text
1	149746	A605546	2022-08-21 16:2	August 2022	Trade (TV)	Owner Surrender	Normal	Don	Spayed Female	11 years





# ERD





# Conclusion/Proposals

- 1). Drawing conclusions from our questions we solve with our data analysis/storytelling.
- 2). Propose ideas on how AAC could improve adoptability by targeting animals that are most adoptable to get them out of the shelter faster.
- 3). Propose new adoption programs that can improve adoption rates of older animals, lower cost upfront, extra supplies, and more.
- 4). Show pers how many pets are available in a local shelter in Austin.